Plánování v robotice

<u>Obecně</u>: Plánování ~ vytváření posloupnosti <u>akcí</u> k dosažení požadovaného cíle (<u>stavu</u>) robotu.

Elementární komponenty plánování:

1. <u>Stavový prostor</u> – vyjadřuje všechny možné situace (stavy), které mohou pro daný systém nastat (např. pozice a orientace robotu, pozice a rychlost (3D) helikoptéry, atd.)
   - spojité a diskrétní verze
   - nekonečné a spočetné/nespočetné

   – velikost stavového prostoru (počet stavů nebo kombinatorická složitost) jsou často příliš veliké k explicitnímu popisu

   – definice stav. prostoru úzce souvisí s užitým plánovacím algoritmem

2. <u>Časová doména</u> – plánovací problém vždy zahrnuje čas jako přirozené uspořádání posloupnosti akcí; 2 typy vyjádření:

- <u>explicitní čas</u> – konkrétní akce v určitém čase (plánování pro jedoucí auto v reálném čase)
- <u>implicitní čas</u> – pouze existence uspořádání akcí (řešení Rubikovy kostky)
- čas může být diskrétní i spojitý

3. <u>Akce</u> – samotný plán je tvořen <u>akcemi</u> určenými k ovlivnění (změně) stavu systému. V plánovací terminologii je <u>akce</u> výsledkem aplikace <u>operátoru</u> a je popsána v plánovacím postupu a odpovídá:

- <u>stavové funkci přechodu</u> (diskrétní) (nabývá hodnot stavů)

  $s_1, s_2 \in S$; $f(s_1, u) = s_2$

  kde s_1 je přípustný pro f

  u ... vstup

- diferenciální rovnice (spojitý případ)

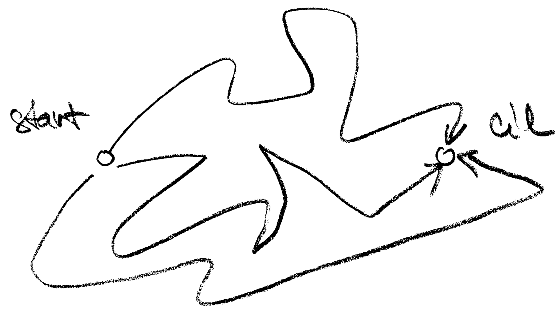
  $s_1, s_2 \in S$; $s_2 = s_1 + \frac{ds_1}{dt} \Delta t + u$

  $\dot{s}_1 = f(s_1)$

- v některých případech se k volbě akcí plánovacím mechanismem přimíchávají akce náhodné ⇒ nutnost řešit plánování za podmínek nejistoty

4. Počáteční a cílový stav (stavy) definují podmínky poč./konc.
   - počáteční (startovací) stav
   - cílový stav (nebo jejich množina)

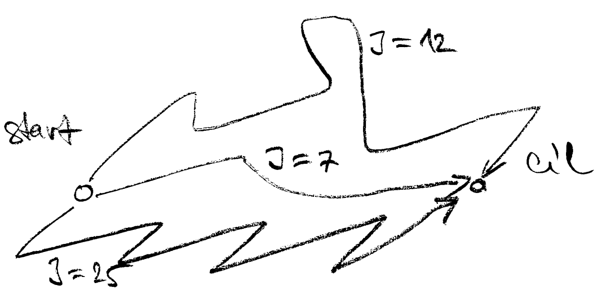
5. Plánovací kritérium (účelová funkce)
   - definuje požadované chování plánovacího mechanismu; vzhledem k čemu bude vytvořený plán optimální.
   - 2 základní přístupy:
     - připustnost plánu
       - hledání takové posloupnosti akcí, která vede do cílového stavu bez ohledu na cenu cesty



$\exists\ f_1, f_2, \dots f_n = f;\quad s_{start} \xrightarrow{f} s_{cíl}$

kde $f_i \in F$

     - optimalita plánu
       - nalezení takové připustné posloupnosti akcí, která vede do cílového stavu při minimalizaci (maximalizaci) vhodné účelové funkce J



oproti předchozímu tedy podmínka navíc $J = \min_{f \in F} J$

- častá užití času potřebného k vykonání plánu, spotřeba energie atd. jako účelová fce J.

- obecně může být nalezení vhodné fce J obtížné

- vhodně komponovaná fce J může zahrnout nejistotu (pravděpodob-nostní popis chování systému) do plánovacího mechanismu

- řešení úloh (hledání plánu) z hlediska přípustnosti je často spojeno s určováním nejhorší možné varianty, která může nastat (generováním všech přípustných plánů a následným hodnocením)

6. Plán - reprezentuje stradegii rozhodovacího postupu (často dáno tzv. účelovou funkcí)

  - jednoduchý plán je posloupnost akcí
  - v případě nemožnosti predikce budoucích stavů je plán dán jako funkce stavu. Plán je pak deterministický
  - v opačném případě je možné realizovat tzv. reaktivní plánování

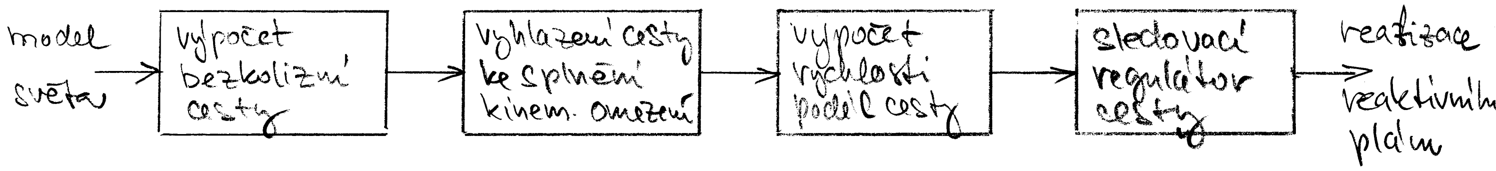
    Plán je nedeterministický

Jak lze s plánem nakládat?

Existující plán lze:

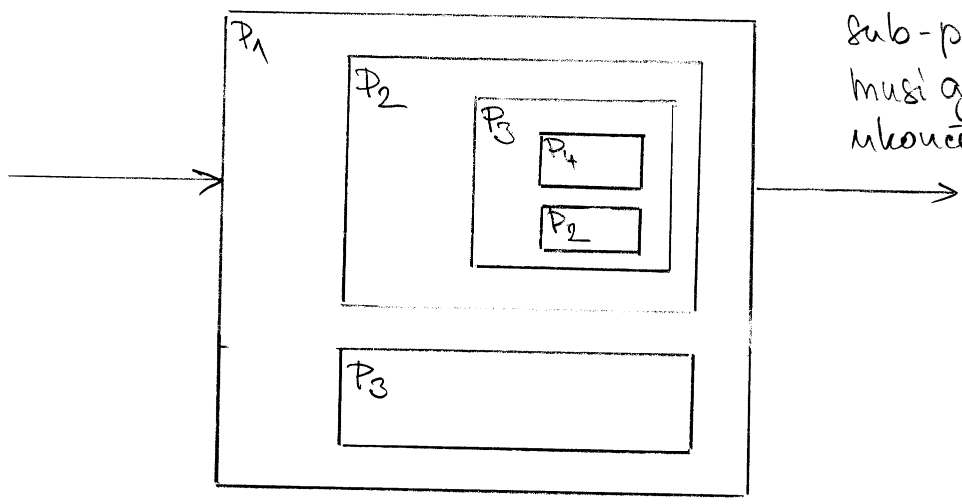
1) realizovat – vykonat určenou posloupnost akcí
2) zlepšit – zdokonalit průběh vykonávání plánu z hlediska účelové funkce nebo jej rozpracovat podrobněji
3) využít v hierarchické struktuře – použít plán jako jednotlivou akci do vyšší úrovně plánování

Zlepšování plánů – př. plánování trajektorie mobilního robotu



- Zatímco předchozí postup realizuje top-down zpřesňování plánů, hierarchická inkluze je bottom-up postupem pro vytváření podrobných plánů

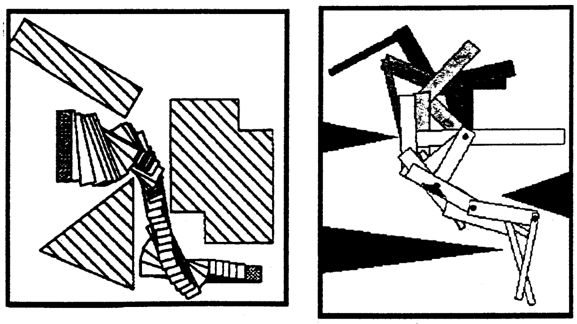
př.



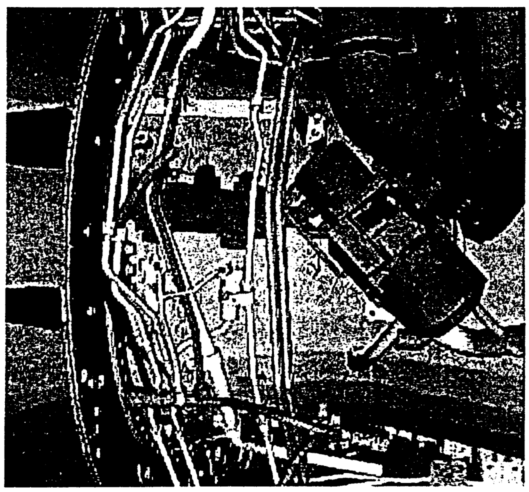
sub-plány musí garantovat ukončení činnosti!

Aplikace plánování

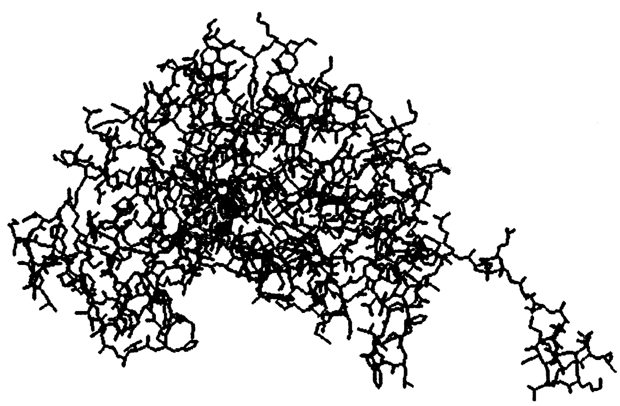
1. Plánování trajektorie robotu



2. CAD návrhy – výměna součástky, dílu



3. Syntéza chemických sloučenin – umístění komponenty sloučeniny do struktury

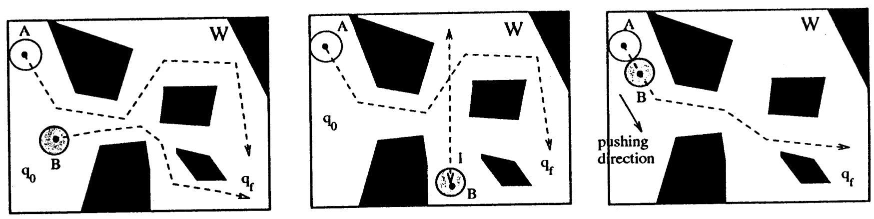


Rozšíření základního problému plánování:

- o pohyblivé objekty prostředí (nejistá poloha objektů nebo objekty v pohybu)
- o více robotů
- o neúplná znalost o prostředí (geometrie, snímání, atd.)
- o neholonomní omezení

příklady:

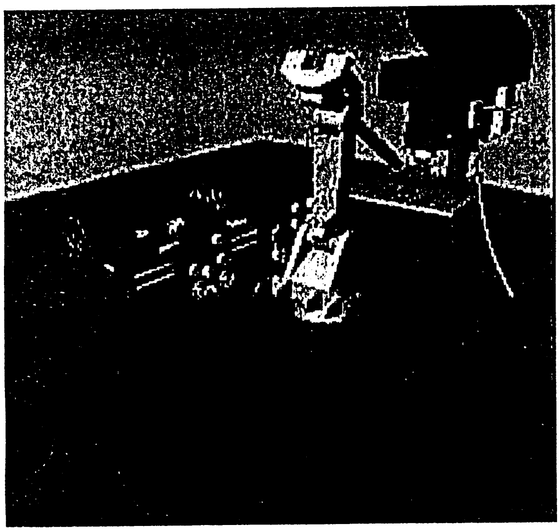
- o montážní robot (plánování manipulace)
- o určování pořadí montáže (pořadí montáže dílů)
- o autonomní vozidla (neholonomní omezení)

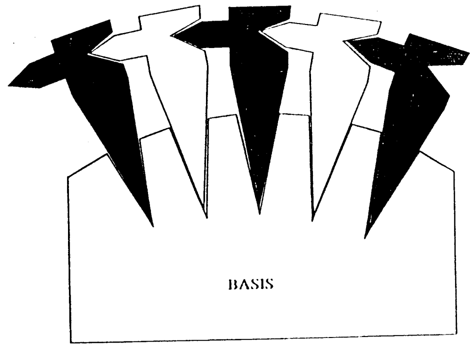


(a) B is a moving robot

(b) B is an obstacle moving on a vertical line

(c) B is an object pushed by the robot





Konfigurační prostor

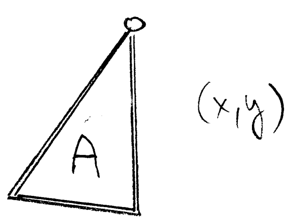
Idea: Unifikovaná reprezentace k-dimenzionálního robotu (robotu s k-stupni volnosti) v parametrickém prostoru.

- do prostoru lze promítnout kinematická a jiná omezení (překážky) a realizovat prohledávací mechanismus k nalezení cesty: start → cíl

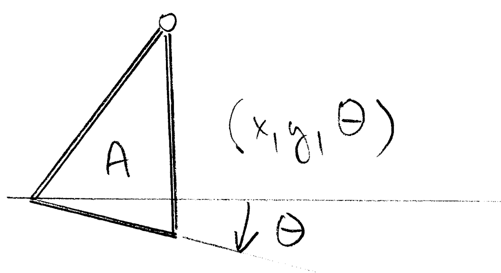
Reprezentace robotu je jako bod (k-dimenzionální) v parametrickém (konfiguračním) prostoru (C-space)

- konfigurace robotu A je reprezentována polohou <u>všech bodů robotu</u> relativně k fixovanému počátku souřadného systému

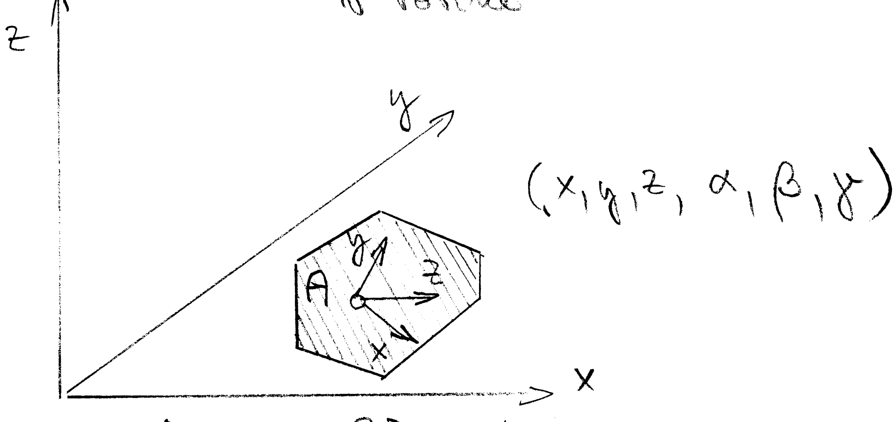
př:



translace robotu v rovině



translace a rotace robotu v rovině



translace a rotace robotu ve 3D prostoru

Zavedíme notaci:

- pracovní prostor robotu: W (workspace)
- robot: A
- konfigurace: q
  - je vektorem parametrů určujících polohu A ve W
  - stupeň volnosti A je k, jenž odpovídá počtu parametrů nezbytných pro určení polohy
- konfigurační prostor: C (C-space)
  - je množinou všech možných konfigurací
- říkáme že:
  - oblast z W je obsazena robotem A pro konfiguraci q: $A(q)$
  - poloha bodu $a \in A$ z W jestliže A je pro q: $a(q)$
- definujeme překážku B takovou, že $B \subset W$ a B je nepřístupnou nebo nedovolenou oblastí C

konfigurace robotu (typy) mohou být:

(1) <u>volná konfigurace</u> (free configuration)

- robot a překážky se nepřekrývají
- $A \cap B = \emptyset$; nemají ani společnou hranici (C-space je tvořen otevřenými množinami)

(2) <u>kontaktní konfigurace</u> (contact configuration)

- robot a překážka se dotýkají
- $A \cap B = \emptyset$; mají pouze společnou hranici (bod)

(3) <u>zablokované konfigurace</u> (blocked configuration)

- robot a překážka se překrývají
- $A \cap B \neq \emptyset$

- předchozí klasifikace rozděluje C-space na 2 třídy: C_{free} a $C_{obst.}$

- kontaktní typ konfigurace lze dle povahy úlohy přiřadit buď k C_{free} nebo C_{obst} ; obvykle však k C_{obst}

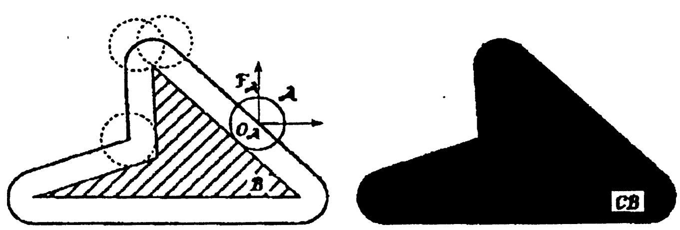
Překážka v C-space

- překážky v prostoru W odpovídají nepřístupným oblastem (C-obstacles) v C-space

$$\text{překážka } CB = \{ q \mid A(q) \cap B \neq \emptyset \}$$

- pro každou překážku $B_i \in W$ existuje odpovídající C-obstacle CB_i, která reprezentuje <u>všechny konfigurace</u> robotu A, jenž kolidují s B_i

<u>Příklad:</u> Mějme kruhový robot A a prac. prostor $W = R^2$. CB polygonu B je získána nárůstem B o poloměr robotu A.



- Více možných významů volných a nepřístupných konfigurací:
  - tradiční význam pro plánování cesty bez kolize (nebo s existencí kolize)
  - robot je příliš daleko (blízko) od překážek (tedy některé bezkolizní trajektorie jsou nepřípustné po zahrnutí rozměru robotu)
  - konfigurace molekuly má příliš nízkou nebo vysokou potenciální energii

Cesta v C-space

- cesta v C-space je "část" spojité křivky propojující 2 konfigurace q_{init} a q_{goal}, tedy cesta je "spojitou mapou":
$$\tau : s \in [0,1] \rightarrow \tau(s) \in C$$
kde: $\tau(0) = q_{init}$ je počáteční konfigurací
$\tau(1) = q_{goal}$ je cílovou konfigurací

- "spojitá mapa" znamená že:
$$\forall s_1, s_2 \in [0,1]; \lim_{s_1 \rightarrow s_2} d(\tau(s_1), \tau(s_2)) = 0$$
kde $d : C \times C \rightarrow R^+ \cup \{0\}$ je <u>metrika</u> v C

<u>Příklad</u>: metrika v C
$$d(q_1, q_2) = \max_{a \in A} \| a(q_1) - a(q_2) \|$$
kde $\|x-y\|$ je Euklidovská metrika ve W □

- další možná omezení na cestu v C
  - minimální délka, hladkost křivky, počet zatáček, omezení max. křivosti, atd.
  - trajektorie může být parametrizována v čase → → přidáním další (časové) dimenze C-space

Druhy cest: volná cesta (free path)

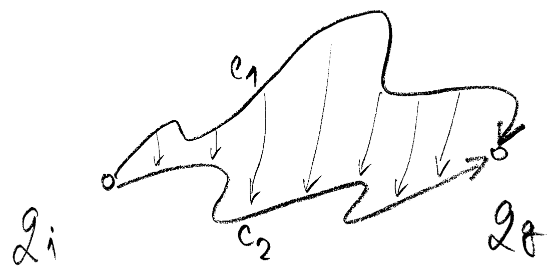
$\tau : [0,1] \to C_{free} = C \setminus CB$

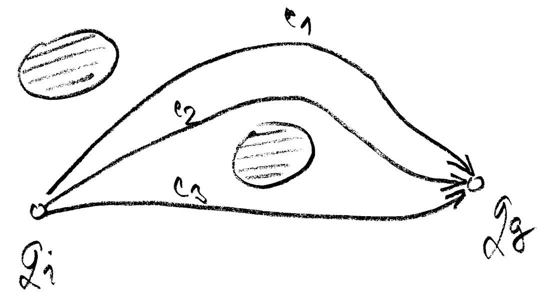
částečně volná cesta (semi-free path)

$\tau : [0,1] \to cl(C_{free})$

může mít kontakt ale není nepřípustná

Homotopické cesty $\overset{df}{\Longleftrightarrow}$ 2 různé cesty mající shodnou počáteční a cílovou konfiguraci a které mohou být spojitě deformováním jedna ve druhou





- homotopické cesty mohou být z hlediska řešení úlohy identické (bez ohledu na "cenu" cesty)

Souvislost C-space

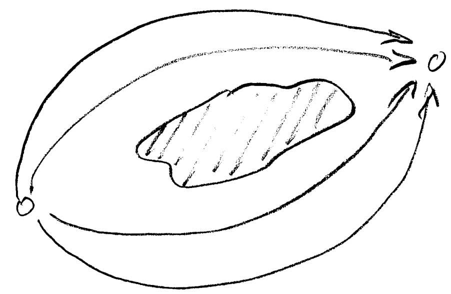
– vlastnost důležitá z hlediska "dosažitelnosti" řešení

Df.: C je <u>souvislý</u> $\overset{df}{\Longleftrightarrow}$ pro všechny možné páry konfigurací $[q_i, q_j]$ existuje vždy aspoň jedna cesta $\tau : q_i \xrightarrow{\tau} q_j$

Df.: C je <u>jednoduše souvislý</u> $\overset{df}{\Longleftrightarrow}$ jakékoliv 2 cesty spojující $[q_i, q_j]$ jsou homotopní

– ostatní případy jsou <u>mnohonásobně souvislé</u>, tj. existuje více než jedna třída homotopie cest

př.:



objekt můžeme obejít zleva nebo zprava ⇒ 2 třídy

Proč je užití C-space výhodné?

- redukce složitosti řešení pro roboty s fyzickým rozměrem (mnoho omezujících podm.) v Euklidovském prostoru → náhrada situací s vícedimenzionálním prostorem (C-space) a bodovým robotem

- C-space je unifikovaný rámec vhodný k porovnávání (vyhodnocování) různých algoritmů

- jednodušší plánování bodových trajektorií

Nevýhody: plánování pohybu je z principu spojité (dáno definicí C-prostoru)

řešení:
- diskretizace plánovacího prostoru C-space
- diskretizace trajektorie
- oba předchozích

Složitost výpočtu plánování cesty

- úplné plánování je výpočetně náročné

úplné plánování ~ plánovač buď nalezne přípustnou cestu nebo oznámí, že řešení neexistuje

⇓

časté použití aproximujících postupů, které naleznou "nějaké" přípustné řešení (neoptimální), jehož dosáhnou v podstatně kratším čase

nebo

nenaleznou řešení žádné (pak nelze ale tvrdit, že řešení neexistuje)

- Obecně pro úplné postupy platí, že výpočet je:
    - exponenciálně náročný s dimenzí C-space
    - polynomiálně náročný se složitostí C_{obst} překážek v C-space (počet stěn objektů, řád jejich algebraického vyjádření)

- Existují 2 ~~zá~~ základní metody obecného plánování (úplného)

(1) <u>úplná dekompozice prostoru</u> (exact cell decomposition)

- dvojitě exponenciální náročnost

$\sim 2^{2^d}$, kde d – dimenze prostoru

- založeno na principu rozkladu prostoru
- C_{free} na jednotlivé oblasti (= elementární buňky) a následné reprezentaci konektivity mezi nimi (graf sousednosti)

(2) <u>metoda cest</u> (roadmap method)

- jednoduše exponenciální náročnost

$\sim 2^d$, kde d – dimenze prostoru

- založeno na výpočtu "siluety" prostoru C_{free}. Reprezentuje konektivitu v C_{free} grafem ve formě sítě 1D křivek

- předchozí platí pro <u>úplné plánování</u>, praxe se snaží úlohu zjednodušit:
  - zjednodušení geometrie robotu/překážek – aproximace
  - omezení počtu stupňů volnosti – dimenze prostoru
  - zjednodušení cest (po částech lineární atd)

tj. plánování se provádí ve 2 krocích:

(1) určení konektivity volného prostoru C_{free} a její reprezentace grafem (nebo funkcí)

(2) hledání výsledné cesty v grafu (funkci)

<u>základní plánovací techniky</u>:

- dekompozice prostoru ✓
- metoda cest ✓
- potenciálové pole

→ je určena funkce s vlastnostmi potenciálového pole $\nabla^2 f(\vec{x}) = 0$ (Laplaceova podmínka) a která má globální extrém v místě cílové konfigurace. Určení (optimální) cesty ~ ~ sledování (největšího) gradientu potenc. pole.

- výpočetní náročnost

$\sim l$, kde l je délka cesty (počet kroků v diskr.)